Software Architecture Document
For
Digitalized System to Book an Ambulance

Version 1.0 approved

Prepared by Debasrito Lahiri (1729122)

11/09/2019

REVISION HISTORY

DATE	VERSION	DESCRIPTION	AUTHOR
11/09/2019	1.0	First Edition	Debasrito Lahiri

Table of Contents

1. Introduction 4
   - 1.1 Purpose 4
   - 1.2 Scope 4
   - 1.3 Definitions, Acronyms, and Abbreviations 4
   - 1.4 References 4

2. Architectural Representation 4

3. Architectural Goals and Constraints 4

4. Use-Case View 5
   - 4.1 Use-Case Realizations 5

5. Logical View 6
   - 5.1 Overview 6
   - 5.2 Architecturally Significant Design Packages 6

6. Process View 7

7. Deployment View 7

8. Implementation View 8
   - 8.1 Overview 8
   - 8.2 Layers 8

9. Size and Performance 8

10. Quality 8

Software Architecture Document

1. Introduction

1.1 Purpose

This document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system.

1.2 Scope

The various architecture designs shown in this document affect the functioning of the whole system. Each module is related to other modules directly or indirectly. As such this document can be considered as an overview of the whole system.

1.3 Definitions, Acronyms, and Abbreviations

JSON: JavaScript Object Notation

Database: Google firebase service

Maps: Google maps platform

HTTP: Hyper Text Transfer Protocol

Environment: Platform of implementation

1.4 References

[1]. Firebase, Wikipedia, 09/08/2017, URL: https://en.wikipedia.org/wiki/Firebase

[2]. Google APIs, Wikipedia, 31/08/19, URL: https://en.wikipedia.org/wiki/Google_APIs

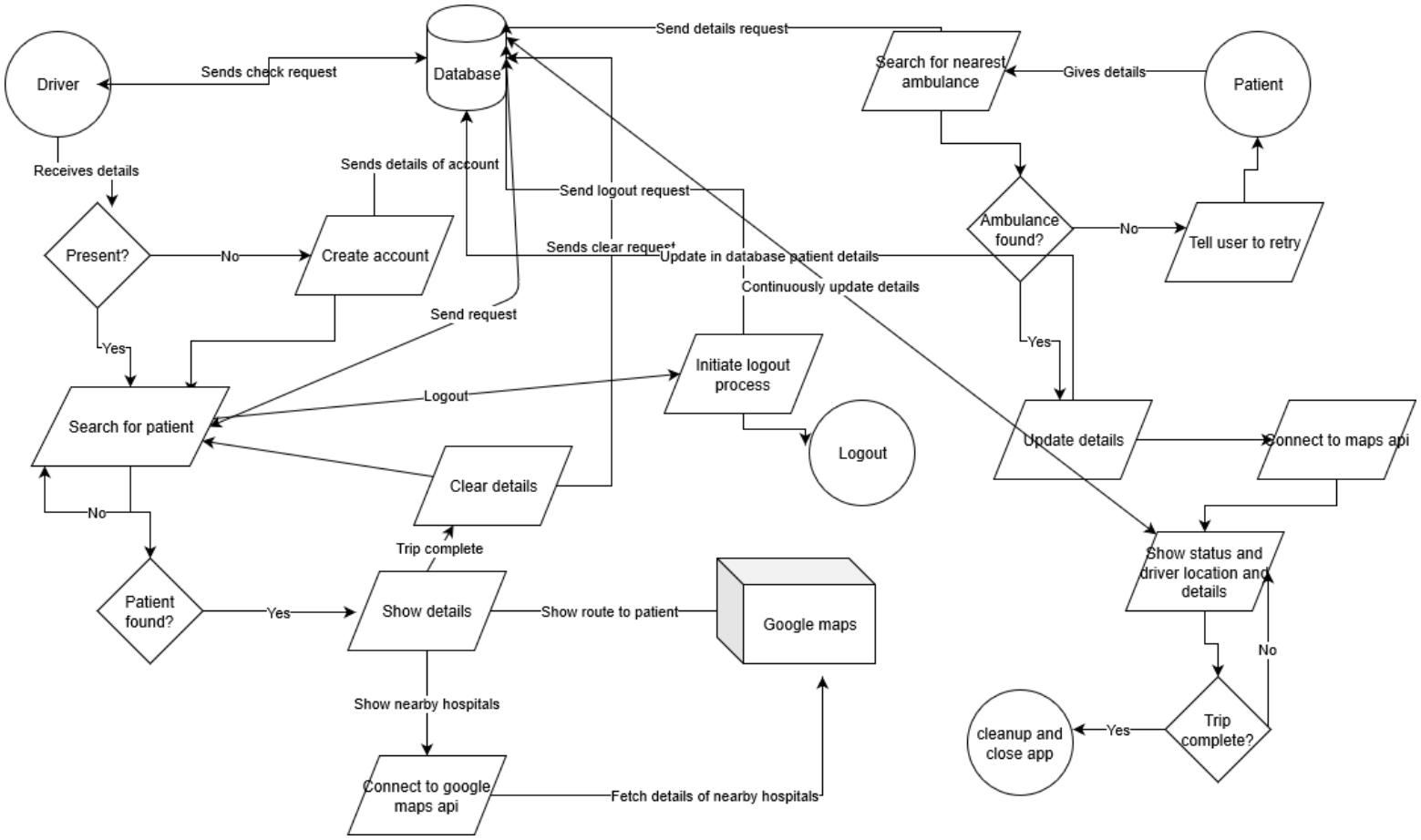
2. Architectural Representation

In this document the relationships between various components, processes, data of the system is shown. The dataflow can also be realized from this document.

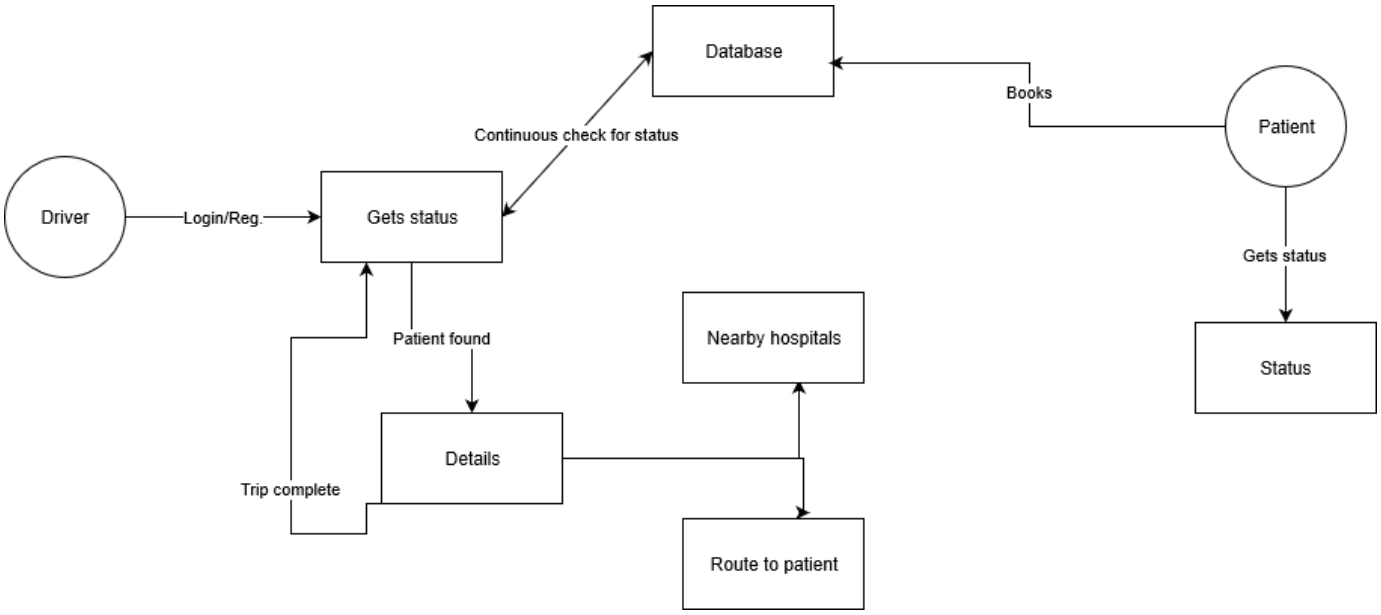
3. Architectural Goals and Constraints

Our system is dependent on real-time databases. Constructing this system on other databases will hamper the system's functioning. Also, the system is dependent on google maps platform. Non-availability of google maps app and/or browser will impact the performance of the system. The connection to the database is real-time, both way and simultaneous for both the apps.

4. Use-Case View



4.1 Use-Case Realizations

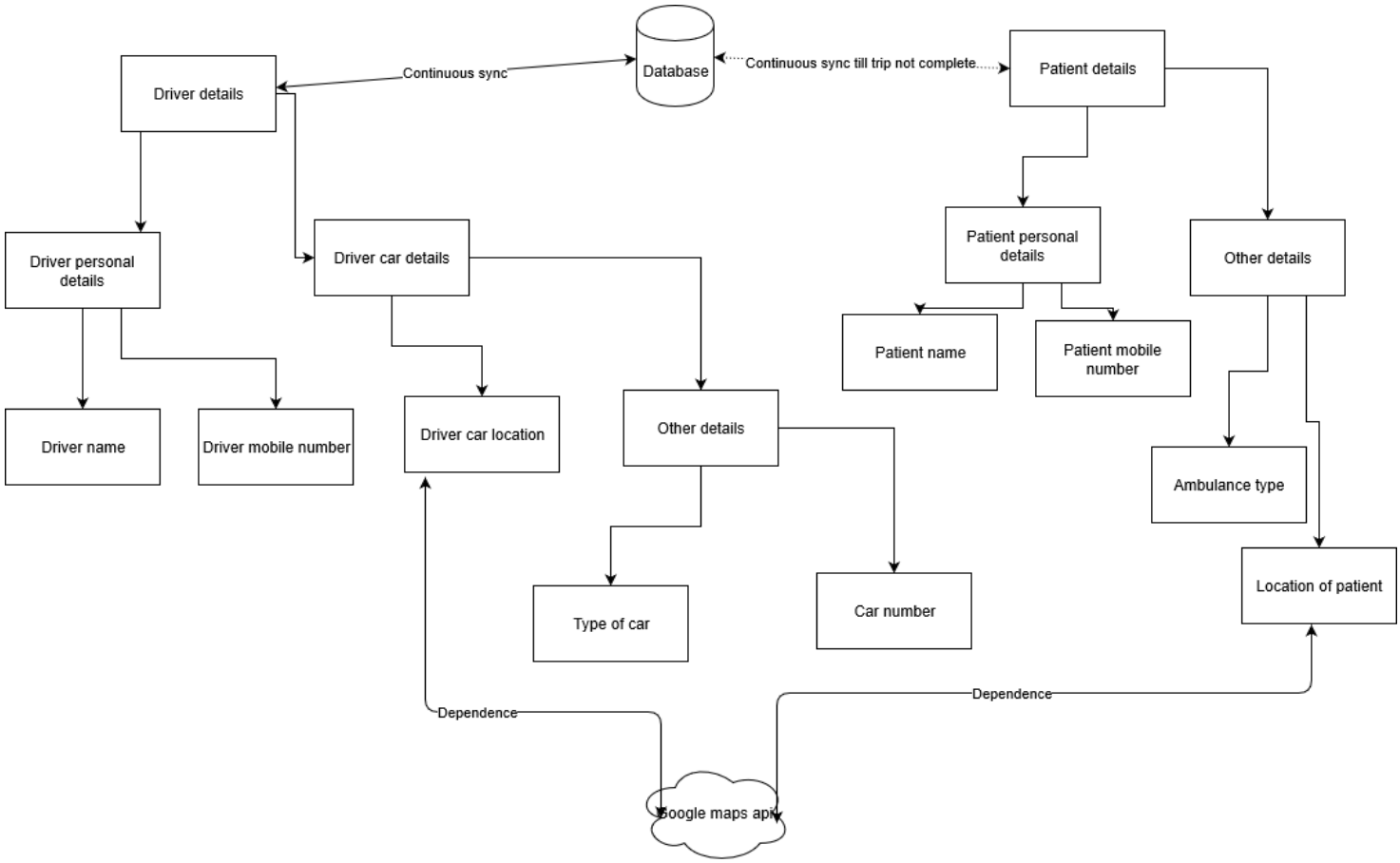


5. Logical View

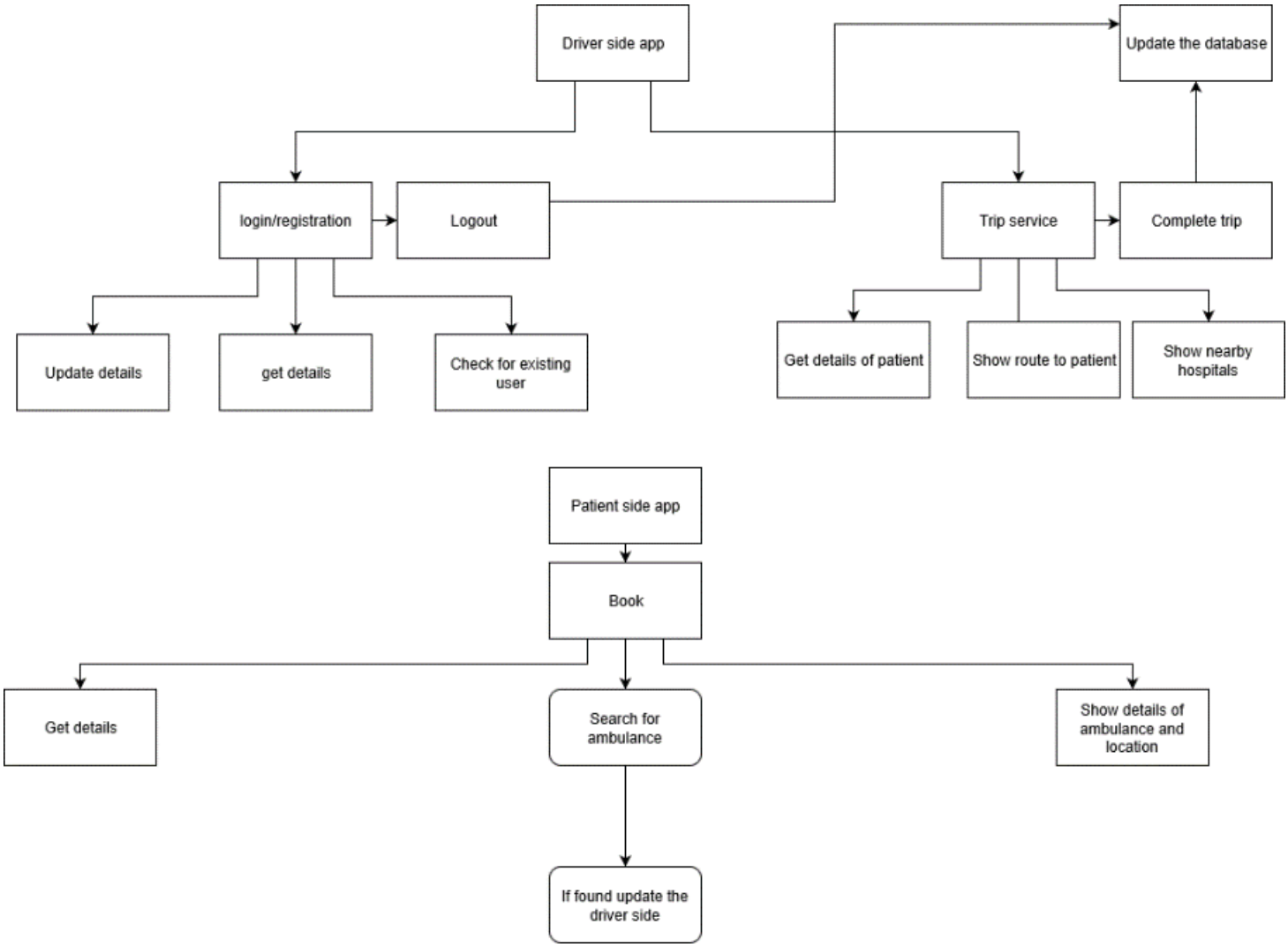
5.1 Overview

Our system consists of mainly 2 data classes and 2 services. The services are Firebase database and Google maps platform. The data classes are Driver data class and patient data class.

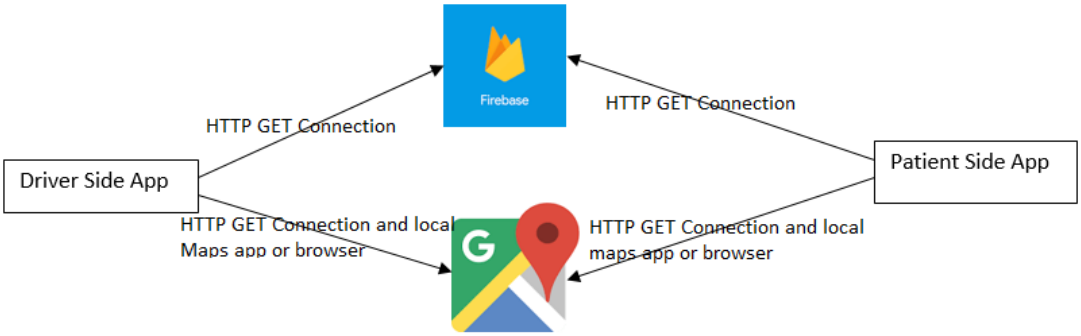
5.2 Architecturally Significant Design Packages



6. Process View



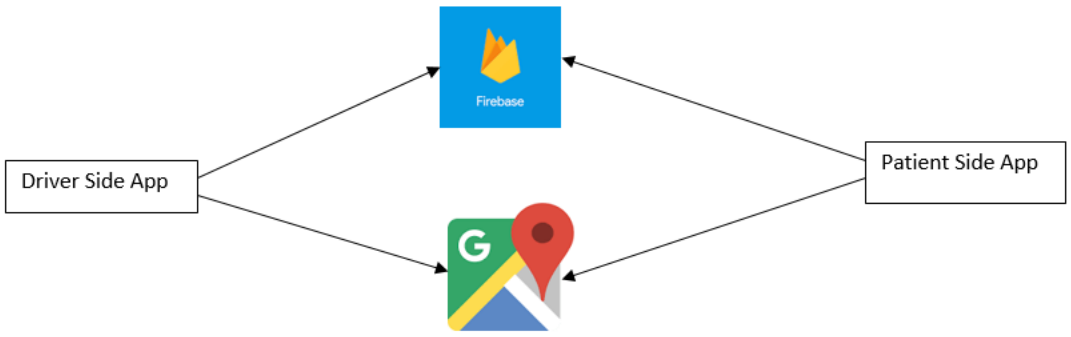
7. Deployment View



8. Implementation View

8.1 Overview

There are 4 components in the first layer of our system. Each of these components have further layers.



8.2 Layers

- Layer 1:

In this layer the basic communication between the remote parts of our system occur.

The four components are:

1. Driver app: It connects to the database and consists of another layer of processes to sync with the database and maps platform.
2. Patient app: This has another layer which connects to the database and maps platform.
3. Firebase: The real-time database used.
4. Google maps platform: The maps platform used for location related tasks.

9. Size and Performance

This system is targeted for mobile devices. As such, implementing it in a device which is not mobile (e.g. a Desktop computer) will result in improper use of the system. Also, the quality of net, device performance etc. will affect the performance of the system.

10. Quality

The architecture of the system is designed in such a way that the system can be ported from one device to another which may or may not contain the same environment as the previous. Such portability is facilitated hugely by the use of firebase database and its integration with Java as well as JavaScript platforms. Also, due to integration of Google maps platform and its support for HTTP Protocols and JSON files, it can be implemented easily in any platform.